REMARKS

Claims previously in issue:	1, 2, 4-11, 13-20, 22-27
Claims amended:	1, 2, 10, 11, 19, 20
Claims canceled:	None
Claims added:	28-39
Claims rejected	1-2, 8-11, 17-20, 26-27
Claims indicated as allowable:	4-7, 13-16, 22-25, if rewritten in independent form
Claims remaining in issue:	1-2, 8-11, 17-20, 26-39

Reconsideration and allowance of the above-referenced application are respectfully requested.

Allowable Subject Matter

Claims 4-7, 13-16, and 22-25 were objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form.

Multiple dependent claims 4, 5, 13, 14, 22, and 23 have been rewritten in independent form as claims 28-39. Accordingly, Applicant submits that claims 28-39 are allowable.

Claim Rejections – 35 USC §103

Claims 1-2, 8-11, 17-20, 26-27 were rejected under 35 U.S.C. §103(a) as being allegedly obvious in view of Iwasawa et al. 5,151,991. Applicant respectfully traverses this rejection with respect to the claims in issue.

The invention includes a system, method, and program for parallelizing applications of certain script-driven software tools. In the illustrated embodiment, scripts in the software tool scripting language are automatically analyzed in order to produce a specification for a parallel computation which is functionally equivalent to the original script. This is accomplished by (a) parsing the script into statements; (b) constructing a serial dataflow graph from the parsed statements; and (c) constructing a parallel dataflow graph from the serial dataflow graph.

The parallel computation specification is then executed by a parallel runtime system, which causes multiple instances of the original software tool and/or supplemental programs to be run as parallel processes. The resulting processes will read input data and produce output data, performing the same computation as was specified by the original script. The combination of the analyzer, runtime system, original software tool, and supplemental programs will, for a given script and input data, produce the same output data as the original software tool alone, but has the capability of using multiple processors in parallel for substantial improvements in overall "throughput".

Iwasawa teaches a method and system for recompiling serial procedural code (e.g., FORTRAN code) in order to parallelize inner and outer loop constructs. See Iwasawa, col. 1, l. 50 to col. 2, l. 13. The loops are taught as repetitive executions of the same statements with possibly differing parameters at each execution (e.g., DO loops).

Iwasawa is thus narrowly focused on a particular type of parallelization. Contrary to the Examiner's assertions, Iwasawa teaches nothing about generation or construction of dataflow graphs (and Applicant certainly did not admit that the Iwasawa reference discloses the claimed limitations). Dataflow graphs comprise a set of vertices coupled by directed edges, or links (see, e.g., Fig. 3 of the application). The vertices represent datasets, processing steps, and intermediate results passed from one step to another, either explicitly or implicitly; see p. 7, ll. 15-19. One of ordinary skill in the art would understand that Iwasawa does not convert the procedural code that it analyses into dataflow graphs. Iwasawa simply detects loops and assigns particular loops to particular processors. Nothing in Iwasawa teaches or suggests a way of solving the completely different problem of parsing a script into statements, constructing a serial dataflow graph from the parsed statements, and constructing a parallel dataflow graph from the serial dataflow graph. Moreover, nothing in Iwasawa teaches analyzing the parallel dataflow graph to generate script fragments in a form that enables a script-driven software tool to execute some of the processing steps.

Method claim 1 and counterpart system and program claims have been amended to clarify that they are directed ultimately to constructing parallel dataflow graphs from serial

dataflow graphs derived from a script comprising at least processing steps and dataset definitions. This is quite distinct from replacing loop constructs with similar loop constructs optimized for parallel processing, as in Iwasawa. Thus, for example, Claim 1 now recites:

a) parsing the script into statements comprising at least processing steps and dataset definitions [support may be found for such language at p. 6, ll. 8-9];

b) constructing a serial dataflow graph from the parsed statements, the serial dataflow graph having nodes connected by directed edges, the nodes representing datasets, processing steps, and intermediate results [support may be found for such language at p. 17, ll. 15-18]; and

c) constructing a parallel dataflow graph from the nodes of the serial dataflow graph such that the parallel dataflow graph may be executed by a parallel runtime system [support may be found for such language at p. 3, ll. 11-15].

Further, method claim 2 (and corresponding system and program claims) has been amended to positively recite the step of:

"analyzing the parallel dataflow graph to generate script fragments in a form that enables the script-driven software tool to execute some of the processing steps."

Nothing in Iwasawa teaches or suggests such analysis or generation of script fragments.

Enclosed is our check in the amount of $2247 including the filing fee of $770 for the Request for Continued Examination, $475 for the 3-month extension fee, and excess claims fee of $1002.

Please apply any other charges or credits to deposit account 06-1050.

Respectfully submitted,

Date: February 26, 2004

Kenyon S. Jenckes
Reg. No. 41,873

PTO Customer No. 20985
Fish & Richardson P.C.
12390 El Camino Real
San Diego, California 92130
Telephone: (858) 678-5070
Facsimile: (858) 678-5099

10369958.doc